DOCUMENT RESUME

ED 093 287 IR 000 768

AUTHOR Groner, Leo H.; Goel, Amrit L.
TITLE Data Management Systems for Structured Information Retrieval. Interim Report.
INSTITUTION Rome Air Development Center, Griffiss AFB, N.Y.
REPORT NO RADC-TR-73-410
PUB DATE Feb 74
NOTE 49p.

EDRS PRICE MF-$0.75 HC-$1.85 PLUS POSTAGE
DESCRIPTORS Computer Programs; Cost Effectiveness; *Data Bases; *Data Processing; *Digital Computers; *Information Retrieval; Program Costs; *Program Design; Technical Reports
IDENTIFIERS Griffiss Air Force Base; *Reliability Analysis Center

ABSTRACT

In this study the advantages and problems involved in using Generalized Data Management Systems (GDMS) to expand the capabilities of information retrieval (IR) systems are investigated. In particular, the design of a Data Manager-1 (DM-1) based system for the Reliability Analysis Center (RAC) at Griffiss AFB, Rome, New York is studied. A design methodology involving a cost criterion is developed. Total system cost and use cost components for each potential system output is explored. System cost is then obtained as a function of these cost components and user demand for outputs. This methodology is then applied to a set of design problems relevant to GDMS/IR data base design. A proposed data base for RAC embodying the solutions to these problems is then given. (Author)

BEST COPY AVAILABLE

ED 093287

RADC-TR-73-410
Interim Report
February 1974

DATA MANAGEMENT SYSTEMS FOR STRUCTURED INFORMATION RETRIEVAL

Syracuse University

Approved for Public Release.
Distribution Unlimited.

Rome Air Development Center
Air Force Systems Command
Griffiss Air Force Base, New York

IR 000 768

DATA MANAGEMENT SYSTEMS FOR STRUCTURED INFORMATION RETRIEVAL

Leo H. Groner*
Amrit L. Goel**

Syracuse University

Approved for Public Release.
Distribution Unlimited.

*Research Assistant, Systems and Information Science.

**Associate Professor, Department of Industrial Engineering and Operations Research; and Systems and Information Science.

Do not return this copy. Retain or destroy.

U S DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

FOREWORD

This interim report has been prepared by Syracuse University under Contract F30602-72-C-0281, Job Order No. 55810213. Mr. Rocco F. Iuorno (ISIM) was the RADC project engineer.

The Office of Information (RADC) has reviewed this report and approved it for release to the National Technical Information Service.

This report has been reviewed and is approved.

APPROVED:

ROCCO F. IUORNO
Information Management Sciences Section
Information Processing Branch

APPROVED:

RICHARD H. THAYER
Colonel, USAF
Chief, Information Sciences Division

FOR THE COMMANDER:

CARLO P. CROCETTI
Chief, Plans Office

SUMMARY

In this study the advantages and problems involved in using Generalized Data Management Systems (GDMS) to expand the capabilities of Information Retrieval (IR) systems are investigated. In particular, the design of a Data Manager-1 (DM-1) based system for the Reliability Analysis Center (RAC) at Griffiss AFB, Rome, New York is studied. A design methodology involving a cost criterion is developed. Total system cost is decomposed into existence and use cost components for each potential system output. System cost is then obtained as a function of these cost components and user demand for outputs. This methodology is then applied to a set of design problems relevant to GDMS/IR data base design. A proposed data base design for RAC embodying the solutions to these problems is then given.

TABLE OF CONTENTS

1. INTRODUCTION 1

2. SYSTEM COMPONENTS 3

2.1 Generalized Data Management Systems 3

2.2 Information Retrieval 4

2.3 Data Manager-1 6

2.3.1 The User's View of DM-1 7

2.3.2 The Data Administrators View of DM-1 7

2.3.3 The Application Programmer View of DM-1 8

2.4 The Reliability Analysis Center 9

3. SYSTEM DESIGN METHODOLOGY 11

3.1 Dimensions of Service 12

3.2 An Abstract System Evaluation Model 15

4. MODEL APPLICATION TO DESIGN PROBLEMS 19

4.1 Representation of Data 19

4.1.1 Data Item Representation 20

4.1.2 Record Design 24

4.1.3 Logical File Organization 26

4.1.4 Physical File Organization 27

4.2 The RAC Data Base 29

4.3 System Functions 36

4.4 System Implementation 38

5. DISCUSSION 40

BIBLIOGRAPHY 41

1. INTRODUCTION

Current information retrieval (IR) systems yield bibliographic data or text as output. The value of such systems would be considerably enhanced if document content could be retrieved in a structured way so that the content could then be used as input to other processes. The availability of generalized data management systems (GDMS) permits the development of more sophisticated systems which combine the capabilities of both kinds of systems.

The purpose of this study is to investigate design methodologies for information systems (IS) of this type. Thus the investigation centers on the application of a GDMS to a system having a large IR component. A specific application is considered by exploring the requirements of the Reliability Analysis Center (RAC) located at the Rome Air Development Center (RADC), Griffiss Air Force Base, Rome, New York. The procedure followed is to determine the requirements for the IS and then to develop the design using an appropriate criterion.

In this study we will be concerned with a parameterized family of designs meeting a common set of requirement constraints. Employing a GDMS permits easy modification of many of the free parameters of the system to tune system performance. Choosing appropriate values of these parameters requires the designer to apply criteria consistent with his overall objectives. Because of its wide applicability, a cost criterion is frequently chosen. The rigorous application of a cost criterion to system design requires the development of models that project the effect of design decisions on the ultimate system cost.

In the development of such a design tool for a system possessing both GDMS and IR capabilities, we study the salient characteristics of GDMS's in general and DM-1 in particular. We then examine the requirements of RAC and their relationship to current IR techniques. A number of design problems requiring solution in the RAC application are then attacked using the cost model developed. The solutions obtained are then embodied in a proposed data base organization for RAC.

An attempt is also made to evaluate the resulting design from the point of view of the user of the system. The user's primary concerns are viewed as lying in four areas.

a. The range and quality of information services that he receives from the system.

b. The demands that the system makes upon him in terms of time and effort.

c. The response time characteristics of the system.

d. The cost of the system.

These considerations are vital in determining whether a system is worth the effort spent in its implementation.

2. SYSTEM COMPONENTS

A first step toward the development of a design is a proper understanding of the system's component technologies and their capabilities and limitation. The primary component technologies that this study is concerned with is that of GDMS and IR. In this section a brief description of GDMS and IR systems is given. As specific examples, Data Manger-1 (DM-1) and the Reliability Analysis Center (RAC) application are discussed.

2.1 Generalized Data Management Systems

A GDMS consists of data, structure and a set of algorithms for manipulating the data and the structure. It acts as a communication channel between the user community and the data base. Minker [18] characterizes a GDMS as follows: "A data management system is considered generalized when it permits the manipulation of newly defined files and data with the existing programs and systems." Minker later adds other qualifications, "[A GDMS] facilitates reference to data by name and not by physical location," and, "[it] facilitates the expression of logical relations among data items." A well designed GDMS permits users to access and manipulate elements of the data base in a way that is both natural and convenient for them and efficient in terms of its system utilization.

Much of the benefit derived from a GDMS results from the insulation of the people and programs from the data. In conventional systems the structure of the data must be explicitly embodied in each program accessing the data. This limits the application of programs to data whose structure has been defined to them. It also limits the ability to restructure data in response to new needs without modifying every program embodying the old structure. These limitations in the

applicability of programs and the flexibility of the data base impose a rigidity upon conventional data processing systems. While this rigidity is not serious in repetitive well defined tasks such as payroll or inventory control, it is a decided obstacle to the successful performance of systems that must respond to continually changing requirements.

GDMS systems evolved from sequential formatted file systems. This evolution has been in the direction of more complex logical data structures and more complex operations upon them. IR systems have had a parallel, but separate, development.

2.2 Information Retrieval

IR systems are characterized by large descriptor spaces. Many of the components of the descriptor space are binary valued attributes that can take on values of either "relevant" or "not relevant" to a given document. Other descriptors may take values from other domains. Publication date may be an integer, author may have as value a set of human beings. Devising a representation for these descriptors and a mapping from descriptor value sets to sets of documents is the task of designing an IR system.

The use of a closed set of descriptors has several limitations. One of these is the requirement for a skilled interpreter to translate user queries into the vocabulary of the system. The translator must understand not only the IR system vocabulary but must have some understanding of the semantics of the user's statement of his request. Another limitation is the inability of the closed vocabulary to react to new distinctions and syntheses that are endemic to any living body of knowledge.

The advent of the computer permitted more flexible strategies. These ranged from searches of full text, to automatic indexing [12], to browsing via remote access [11]. All of these are attempts to escape the rigidities imposed by the limitations of the fixed descriptor set. Edwards [11] suggests a dialogue between the IR system and the inquirer. Given an initial query the system helps the inquirer reformulate his query by presenting him with alternative index terms "close" to the ones he has used. The dialogue continues until the user feels that the system will interpret his query in a way that he intends. The IBM Document Processing System uses estimates of the number of records to be retrieved by a query as a preliminary indication to the user of the effectiveness of his query in obtaining the response he wants.

Movement away from the rigid hierarchical classification schemes has not eliminated the problem of relationships among descriptors. According to Edwards, "Examples of these relations are synonomy, generic-specific, whole-part, contiguity and autonomy." In contrast to the semantic concept of descriptor relation there are a number of empirical measures. One such measure is relative frequency of co-occurrence defined by Jackson [13].

$$\text{Co-occurrence } (t_j, t_k) = |D_j \cap D_k| \,/\, |D_j \cup D_k|$$

A second measure is correlation coefficient:

$$\text{Correl } (t_j, t_k) = \left(\frac{|D_j \cap D_k|^2}{|D_j| \cdot |D_k|} \right)^{1/2}$$

In both expressions D_i is the set of documents indexed by descriptor t_i and $|D_i|$ is the number of documents in D_i.

The effect of a hierarchical thesaurus is to constrain certain correllations to values of zero or one. This has the effect of constraining the correlations of sister specific terms having a common generic descriptor to be zero. It also has the effect of constraining the conditional probabilities of some generic terms given their specific terms to a value of one. This information on descriptor dependence can be used to encode the descriptor data more compactly. The advantage that can be taken of information contained in the thesaurus depends upon whether one views it as a compendium of immutable truths or as an empirical statement of experience to date.

2.3 DM-1

Data Manager-1 (DM-1) is a system developed by Auerbach Corporation for RADC, Griffiss AFB, Rome, N.Y. As a representative GDMS, DM-1 acts as an interface between the data base and the people and programs that must access it.

The implementer of a DM-1 based system must be aware of the interfaces DM-1 presents to its users and administrators. DM-1 may be classified as a host rather than self-contained GDMS.[1] As a host GDMS, data is accessed via subroutine calls from programming languages. Thus the implementer must also be aware of the DM-1-programmer interface. In the following paragraphs, DM-1 is described as viewed by the user, the administrator and the programmer.

---

[1] There are plans for implementation of DM-1 self-contained capabilities.

2.3.1 The User's View of DM-1

To the user DM-1 provides a means using both the data base and a library of programs in ways that meet his current needs. DM-1 provides methods for structuring data and a set of tools for managing and querying the data base. Capabilities for naming fields, files and defining their structure and relationships are provided. Data may be entered, manipulated, modified and restructured. The user may invoke conditional search, query, dialog and display capabilities for defining significant data, retrieving it for display or further processing to meet his current needs.

DM-1 facilitates the use of programs by providing ways of defining, storing and executing jobs involving one or more programs. Complex jobs may be defined as a sequence of programs with binding specifications to relate data input and output of component programs. Since DM-1 maintains structural information concerning both the data and the input and output of programs, it has the capability to automatically restructure existing data to meet the requirements of the programs. Program output in turn can be restructured as additions or modifications to the data base.

2.3.2 The Data Administrator's View of DM-1

The data administrator is responsible for maximizing the effectiveness of the data base within the constraints imposed by budgets, hardware and security. Although primarily responsible for data items within the data base, the data administrator can encourage sharing of common data by different applications. The differing data structures and formats required by different applications can be satisfied by maintaining multiple files carrying the same data in different forms but only at the costs of multiple updates, extra storage and reconcilliation problems. DM-1 obviates the need for multiple copies of files differing only in format or sequence.

single structure may be selected and transformed through the DM-1 binding and dexing facilities to meet the needs of other applications. The optimal choice data organization requires information concerning the type and frequency of cess to data. For other than purely batch processing applications where accesses y be easily predictable this information involves statistics concerning user havior. Usage statistics gathered by DM-1 can be analyzed to indicate changes in ganization that will make the data system more reponsive to the current pattern usage.

3.3 The Application Programmer's View of DM-1

DM-1 maintains a library of programs. When invoked, these programs use DM-1 cilities to retrieve input data and to store outputs. The programs are independent the external structure of the data. The data is delivered to the program's input ffer in the form the program requires, having been transformed from its storage form. similar transformation of outputs occurs. These tranformations are guided by e data directories and by the program description library. The program is written operate with input and output parameters that need refer only to that part of e data base actually acted upon.

Programs retrieve data items by giving them the formal names specified by the plication programmer. The job description or job request ties the formal name to e actual data base item. The appropriate service routine then fills the programs' ffers with the desired data. Output follows a similar course. The program, erefore, need not be concerned with the structure of any of the data in storage.

Programs to perform statistical analyses, for example, can be written to operate on pure observation data arrays. The DM-1 binding mechanism can be used to select the required observation from complex data records and present it to the statistical program.

The major structural choices made by the data administrator involve the choice of hierarchies and the related choice of data sequence and the determination of whether or not to index data items. Indexing a data item creates an auxiliary table permitting more rapid retrieval of data associated with the values of the indexed item.

2.4 The Reliability Analysis Center

An information system must be evaluated in terms of its satisfaction of user requirements. The user of RACIS is the Reliability Analysis Center (RAC). It is administered by RADC at Griffiss AFB, New York.

The primary objectives of the center are [1]:

a. To upgrade systems reliability through centralized collection, analysis and dissemination of reliability and experience information;

b. To provide better understanding of the nature of failures in microcircuits leading to improved designs and applications; and

c. To reduce costs by eliminating duplicate testing of microcircuit devices by DOD contractors.

In support of these objectives RAC must maintain a data base consisting of microcircuit failure data as well as technical monographs relating to failures. Since a substantial proportion of the failure data appears in technical reports, traceability requirements demand full cross referencing of test data to their document sources. "The center concentrates on the acquisition of information generated under

previous and current device testing and analysis programs, equipment design, fabrication, testing and field test and operational experience. Major emphasis is placed on documentation of all factors leading to failure, such that the effects of design, process control, and applications can be isolated and analysed." [1]

The RAC, with its comprehensive data base, has analysis capability covering microcircuit technologies including: semiconductor integrated circuits, thick and thin film circuits, chip elements, hybrid devices, and large scale arrays (LSI). The Center's resources are continually augmented through systematic acquisition of reliability test and operational data.

Currently RAC uses manual IR techniques based on a Termatrex punched card system. Since its inception, its data base has grown to the point where manual techniques are becoming infeasible. In addition, demands for more elaborate analyses of data and an expansion of RAC coverage to include hybrid and passive devices make the implementation of a computer based system imperative.

The RACIS must be capable of satisfying a wide range of user requirements. It must be capable of performing IR tasks as well as performing statistical analyses of failure data. In a study by Auerbach Corporation it was proposed that DM-1 be employed as a vehicle for the RACIS implementation [1]. G.T. Capraro [4], in 1971, presented a proposal for a one to one mapping of the current manual RAC system into a DM-1 system. This proposal demonstrated the feasibility of the use of DM-1.

3. SYSTEM DESIGN METHODOLOGY

If the process of design is essentially one of making choices then the selection of criteria for decisions implicitly determines the design. In this section a criterion is adopted and an abstract model embodying this criterion developed. Some of the problems in applying this model are also discussed.

A cost criterion has been adopted because of its logical appeal and wide applicability. Furthermore, many of the commonly used criteria can be translated into cost. Reliability, response time and compatibility, for example can be translated into costs of failure, waiting and interfacing respectively. The definition of cost must be broad enough in its scope to capture all of the benefits and losses attributable to the system. Therefore, in this section the overall system cost has been used as the design criterion.

The use of a cost criterion in system design involves the definition of what is to be designed, parameter space within which designs are to be selected and of a cost function which can be used to compare the alternative costs of design decisions. The design space is narrowed by implicit and explicit constraints imposed by requirements and technology. The fixing of each design parameter further constrains the remaining choices available to the designer. The successive decisions made in the course of design may be viewed as a path traced through an extensive decision tree. If a technique existed for associating a cost with each feasible path then the process of system design could be treated as a dynamic programming problem.

The design of any information processing system requires a determination of the level of user activity and its distribution. In addition to stochastic and time varying aspects of user demand, demand may change as a response to the level of information service provided. The demand for RACIS services can be expected to be especially elastic. One can see this clearly if one compares its demand with that for the

services of an airline reservation system. Customers' propensity to reserve seats depends primarily upon their desire to travel and very little on the performance of the information system. The primary product of RAC is information, however, and the demand for RACIS services can be expected to vary considerably with the level of service provided. This dependency of demand upon performance makes the job of estimating user activity more difficult than simply projecting historical demand.

Changes in the pattern of demand may take the form of a change in the overall query arrival rate or they may involve shifts in the type of query. Recognition of these changes requires that the system maintain statistics on queries. To determine the effects of demand changes, system modification and tuning performance statistics are also required. While routines to collect query and performance data must be resident, the data collected can be written out onto a system log for off-line reduction and analysis.

Among the more important decisions these statistics can support is the choice of data items to be indexed. Frequency and joint frequency data on the appearance of data items in the qualification portion of queries can be used to indicate indexes to be added or dropped.

3.1 Dimensions of Service

In addition to response time, the level of service provided by an information system can be characterized along other dimensions User response to changes along any dimension of service will be a function of this perception of the benefits he derives by using the service. As long as there are costs associated with its use he will use the service only if his perceived benefits exceed his cost. This holds whether the costs are direct monetary charges or indirect costs of time and effort.

Some of the dimensions of system service are:

a. The scope of information available through the system.

b. The capabilities to manipulate or process data.

c. Ease of use.

While response time is degraded by increasing demand, information scope and processing capabilities are not. In fact, if clients act as contributors of data and procedures as well as users, a growing community of users may lead to a richer system. If the system is to grow in this way the data and procedures contributed by various groups must be useable by other users in new contexts. It is here that the special capabilities of generalized systems come to the fore. Users differ in their demand for rapid response time scope of information and processing capabilities. Not only are there many kinds of potential users but their needs change with time. There is a second type of response time that concerns the time it takes the system to respond to a demand for new class of data or a new processing capability. Overall system demand is as much a function of the second type of response time as of the first. User impatience with a system that cannot respond in time to his current problems may lead him to give up the system.

Ease of use must be evaluated for different classes of user. Novice users, sophisticated but casual users and regular heavy users will differ in their assessments. One major component of ease of use is the amount that the user must learn or memorise to successfully use the system. Systems requiring consistent reference to supporting documentation receive low scores on this measure. Mnemonic significance can lead to increased bulk of input required and output generated. The optimal balance is determined by the user's expected experience level.

The performance variables that enter into the cost function are functions not only of design but also of user demand distributions, data distributions and in the case of multiprogrammed systems the distributions of events in other programs. The large number of random variables involved make closed form analysis of the performance variable distributions rare except in extremely simple or idealized cases. One approach frequently taken is simulation of certain aspects of the proposed designs. Although systems too complex for closed form analysis have been successfully simulated, there are limitations. The result of a simulation run is usually a sample value from a distribution whose form is often an unknown function of model parameters. Experimental design, and sampling theory can help reduce the number of data points required to obtain response surfaces via regression techniques. Nevertheless, where many levels of many variables are involved and response variance is not negligible the simulation approach becomes unworkable.

The problem of selecting a good, if not optimum, design can be made more tractable by partitioning the overall design problem into smaller sub-problems. The effectiveness of the partitioning approach varies with the degree of interaction between the variables of different partitions. If it can be shown that designs containing a particular subset of decisions are superior to designs not containing that subset, then the optimal design contains that subset. The original problem can be replaced by a smaller one not containing the variables determined.

Where this condition does not hold there is a danger that we will fail to find the optimum in any attempt to fix some of the variables independently of others. If the problem is too large to be solved as a unit we may be willing to tolerate the possibility of suboptimization inherent in a piece-meal approach as the price of getting any design at all.

3.2 An Abstract System Evaluation Model

In defining a system it is important to distinguish between the system and its implementation. A system may be defined in terms of its function. In a given state it yields certain outputs when given certain inputs. An implementation is a means for realizing that function. It is only when a system has been precisely defined that a search for an optimal implementation can begin. These ideas will be formalized below.

A system S may be defined as a 4-tuple $<Q,R,\Sigma,S>$ where Q is a set of possible inputs, R is a set of possible outputs, Σ is a set of states and S is a function

$$S: \quad Q \times \Sigma \rightarrow R \times \Sigma$$

Members of the output set R are viewed as being composed of constituents from a set D. $D \subseteq \{d \mid d: \Sigma \rightarrow J \cup A^*\}$. That is D is a set of functions d, that takes on particular values as integers in J or character strings A^* over an alphabet A at a given state σ in Σ. Thus D is the set of all variables whose values may appear in outputs. An output r from the output set R is an element of the power set of the set of output components D.

The definition of a system given above is independent of implementation considerations. It is desirable to relate the abstract system to its implementations. A concrete state $\hat{\sigma}$ may be defined as a triple $<h,p,m>$ where h is the hardware, p is the programs, and m is the content of all levels of memory including control registers. A concrete system may be defined as a mapping

$$\hat{S}(h,p): \quad Q \times M \rightarrow R \times M$$

We will denote by S_1, and S_2 the mappings into R and M respectively and where there is no ambiguity we will skip the h and p arguments. It must be understood however that

many of the functions given have h and p as implicit arguments.

Let S be a mapping

$$\delta : \hat{\Sigma} \rightarrow \Sigma \quad \text{such that}$$

$$S_1(q,\delta(h,p,m)) = \hat{S}_1(h,p)(q,m) \text{ and}$$

$$S_2(q,\delta(h,p,m)) = \delta(h,p,\hat{S}_2(h,p)(q,m)).$$

That is let δ be a morphism from $\hat{S}$ to S. If the above equalities hold for all $q \in Q$ and $m \in M$ then we will say $\hat{S}(h,p)$ is an implementation of S and that δ is a design for S.

Let $\hat{\delta}$ be a space of implementations for S. Let \$ be the reals interpreted as costs. We may determine functions C_e and C_u regarding the existence and use costs of data over some period of interest. For a given variable $d \in D$ and a given implementation S(H,P), let $C_e(m^*,d)$ be the cost of maintaining d ready for use through the concrete state history m^* where $m^* \in M^*$ is a sequence of memory states. C_e may include costs for gathering information, keypunching, any preliminary processing, any storage space, and required updating and the eventual deletion cost for that variable. It would consist of all costs associated with d even if d were never accessed for use. C_e is given as a function of state history since events in state history affect existing cost. E.g., in some systems data may migrate through the memory hierarchy with storage costs dependent upon usage.

Let $C_u(q,m)$ be the unit cost of processing input q from the system in concrete state m. Use cost may include CPU, channel and peripheral usage, time people spend waiting for results and time spent interpreting results. Let $\nu(q,\sigma,m)$ be the number of times over some time period of interest that input q is received when the system is in concrete state m.

We may then express the cost of an implementation S(h,p) of S with design δ as:

$$C(m^*,\nu) = \sum_{d \in D} C_e(m^*,d) +$$

$$\sum_{q \in Q} \sum_{\sigma \in \Sigma} \sum_{m \in \delta^{-1}(\sigma)} \nu(q,\sigma,m)\ C_u(h,p)(q,m)$$

where $\delta^{-1}(\sigma) = \{m \mid \sigma = \delta(m)\}$

The expression above is a function of state history m^*. In systems where existence cost is dependent on state e.g. multi-level memory hierarchies where data residence depends upon usage, this function may be quite complicated. However, in systems where existence costs are independent of state history:

$$C_e(m^*,d) = C_e(d)$$

and

$$C(\nu) = \sum_{d \in D} C_e(d) +$$

$$\sum_{q \in Q} \sum_{\sigma \in \Sigma} \sum_{m \in \delta^{-1}(\sigma)} \nu(q,\sigma,m)\ C_u(q,m)$$

Note that $Q \times \Sigma \times M$ is finite and treating $\langle q,\sigma,m \rangle$ as indices to vectors, the second term above may be expressed as $\underset{\sim}{\nu} \cdot \underset{\sim}{C}_u$ where $\underset{\sim}{\nu}$ and $\underset{\sim}{C}_u$ are vectors (or tensors) indexed by $\langle q,\sigma,m \rangle$. This notational convenience may be extended to the entire cost expression

$$C(\underset{\sim}{\nu}) = \underset{\sim}{1} \cdot \underset{\sim}{C}_e + \underset{\sim}{\nu} \cdot \underset{\sim}{C}_u$$

At the design stage complete information regarding $\underset{\sim}{\nu}$ is rarely known. If information regarding its probability distribution is available then we may be able to derive the distribution of $C(\underset{\sim}{\nu})$, the total system cost. If $\phi_{\nu(q,\sigma,m)}(t)$ is the characteristic function of ν then

$$\phi_C(t) = \prod_{\langle q,\sigma,m \rangle \in Q \times \Sigma \times M} \phi_{\nu(q,\sigma,m)}(t \times C_u(q,m))$$

The set $R \times \Sigma \times M$ is, of course, quite large and anything that can be done to reduce it to manageable size is worth considering. Substantial reductions are possible by reducing $R \times \Sigma \times M$ to equivalence classes over the cost functions and redefining these functions over the quotient sets.

4. MODEL APPLICATION TO DESIGN PROBLEMS

The abstract model developed above may be applied to many of the problems of systems design. One of the areas in which the concepts of existence and use costs have a natural interpretation is that of device selection for data residence. Here the relevant part of data existence cost is simply portion of device rental associated with the space occupied by the data set. The relevant use cost is a function of the access time and the urgency of the need for the data.

A large part of system design involves software rather than hardware decisions. These choices involve data representation and accessing strategies. Here existence cost involves storage of not just the primary data but also storage of indexes, tables and algorithms as well as the cost of maintaining them and creating them. Use cost must be a measure of the total impact of a query upon the system.

4.1 Representation of Data

One of the important design decisions concerns the representation of data in the data base. Choices here affect disk requirements, buffer sizes, I/O rates and processing requirements. The data representation problem recurs at several levels:

a. At the data item level information must be encoded into a pattern of bits.

b. At the record level a range of options exist between fully formatted, variable format and unformatted records. There are also choices to be made as to distribution of information between records or establishing subsumed files.

c. At the logical file design level choices must be made regarding attribute-object-value relations.

d. At the physical file design level an implementation for the logical file design is determined. The primary consideration here is the ability to access information in the file efficiently.

4.1.1 Data Item Representations

The users of an information system have a right to expect that their interaction with it is in a human rather than machine oriented form. While they are marvelously adaptable creatures, users need not be antagonized by barbarous codes which must either be memorized or laboriously translated by referencing system documentation. Compact codes are desireable for internal system efficiency, however this efficiency need not be at the expense of the user if the system can perform the translation task for him via a codebook.

Consider the document attribute "security classification". It is desireable to communicate values of the attribute in a display form such as "secret" or "unclass". Where the number of possible values of an attribute is bounded by n then it may be represented in several ways:

a. As a bit vector of length n i.e. as n one bit fields each representing the presence or absence of one of the attribute values.

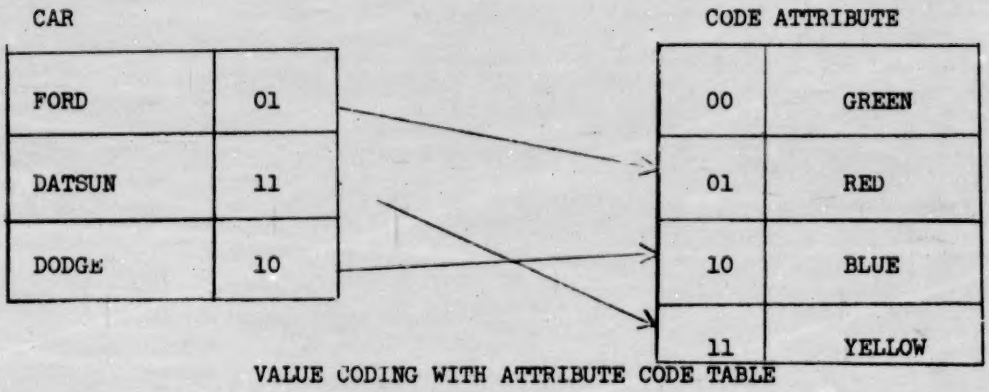
b. As a binary number in the range 0 to n-1. This representation takes $\log n$ bits of storage. This is called value coding.

c. As a set of alphabetic characters giving the value in display form.

The bit vector approach has the advantage of fast comparison for equal values. Its disadvantages are the space required when the attribute can have more then two values, and its inflexibility. Unless bit positions are reserved for new values the introduction of new values will require a reorganization of the file. The bit vector approach records information redundantly in that it does not take advantage of value dependencies that may occur.

CAR	RED	BLUE	YELLOW
FORD	1	0	0
DATSUN	0	0	1
DODGE	0	1	0

BIT VECTOR ENCODING



VALUE CODING WITH ATTRIBUTE CODE TABLE

CAR	COLOR
FORD	RED
DATSUN	YELLOW
DODGE	BLUE

DISPLAY VALUES STORED WITH RECORD

FIGURE 4.1 ALTERNATIVE DATA ITEM REPRESENTATIONS

Value coding requires fewer bits. The saving between it and bit vector for n values is $n - \log_2 n$. In addition while the bit vector approach must explicitly reserve space for new values the value code approach does not use codes in the range $n+1$ to $2^{\lceil \log_2 n \rceil} - 1$, hence these codes are available for new values.

Both techniques require bijective mappings from representations to display values. Mappings to display value from bit position or value code can be implemented by using either argument to index a table of display values. The display value to position or code mapping might use binary search or hash coding into a table of bit positions or value codes respectively.

Using display values in the file requires no encoding or decoding mappings but does usually involve significantly more storage. This storage difference manifests itself not only in the primary file but also in the index if the attribute is indexed. Since in many systems including DM-1 the size of an index block is fixed there is an inverse relationship between the size of the value representations and the number of distinct values that can be accommodated in an index block. The number of levels required in a hierarchical index is $\lceil \log_m s \rceil$ where m is the number of values per index block and s in the number of distinct values of the attribute in the file. Increasing m means fewer levels of index and fewer I/O events per primary record retrieval.

4.1.2 Record Design

If we represent each entity as a record in the file we must decide upon a representation for attributes and values. We have several alternatives:

a. Formatted record. We associate with each attribute a fixed field of sufficient size to accommodate all value representations. If the attribute can take on values of only "relevant" or "irrelevant" then a one bit field will do.

b. Unformatted record. We associate with each attribute a distinct code and we store only in those records where the attribute is relevant the attribute code and value representation. Thus each record consists of a string of attribute-value pairs. If the attribute can take on values of "relevant" or "irrelevant" then only the attribute code need be stored since the value may be inferred.

c. Multi-format records. Where relevancy patterns exist a code may be used to select a format associated with a pattern.

d. Mixed strategy. Some attributes may be associated with fixed positions in the record while others are associated with attribute codes.

We would like to compare these record organizations on the basis of storage space and attribute retrieval time once the record is available. Except in so far as record length is affected none of the organizations discussed prejudices any retrieval technique so they cannot be compared on that basis.

Let there be n entity records E_j and m attributes A_i. Let r_{ji} = if A_i is relevant to E_j then 1 else 0. Let c_i be the space required to store a representation of the value of A_i. Using formatted records for the entire file the space needed for A_i is

$$S_i^f = nc_i$$

NAME	PLACE OF BIRTH	AUTO REG.	YEAR NATURALIZED
LEO GRONER	CRACOW, POLAND	-	1953
HARRIET PERLA	NEW YORK, N.Y.	156732	-
YUAN LIU	SHANGHAI, CHINA	217513	1961

FORMATTED RECORDS

1	LEO GRONER	2	CRACOW, POLAND	4	1953		
1	HARRIET PERLA	2	NEW YORK, N.Y.	3	156732		
1	YUAN LIU	2	SHANGHAI, CHINA	3	217513	4	1961

CODE ATTRIBUTE

1	NAME
2	PLACE OF BIRTH
3	AUTO REG.
4	YEAR NATURALIZED

UNFORMATTED RECORDS WITH ATTRIBUTE CODE TABLE

NAME	PLACE OF BIRTH	UNFORMATTED ATTRIBUTES			
LEO GRONER	CRACOW,POLAND	2	1953		
HARRIET PERLA	NEW YORK,N.Y.	1	156732		
YUAN LIU	SHANGHAI, CHINA	1	217513	2	1961

CODE ATTRIBUTE

1	AUTO REG.
2	YEAR NATURALIZED

MIXED FORMAT STRATEGY WITH ATTRIBUTE CODE TABLE

FIGURE 4.2 FORMATTED, UNFORMATTED AND MIXED RECORDS

If the A_i is stored in unformatted form and b bits are required for attribute code and v_i bits are needed for the value then letting $d_i = b+v_i$ the space needed for A_i throughout the file is

$$S_i^u = \sum_{j=1}^{n} r_{ji} d_i \quad .$$

Note that $c_i - 1 \leq v_i \leq c_i$.

We can consider r_{ji}, $j = 1,2,\ldots,n$ to be n independent identically distributed Bernoulli random variables with parameter p_i. Since $S_i^u = \sum_{j=1}^{n} r_{ji} d_i$, it is binomial with parameters n and p_i. Therefore $E(S_i^u) = np_i d_i$ and var $(S_i^u) = np_i(1-p_i)d_i$.

Retrieval time of an attribute value from a record involves a mapping from the attribute to its value representation's location in the record. For formatted records this mapping is fixed over the set of records in the file. For unformatted records since the location cannot be predicted a search is required to access any value for the attribute sought in every record.

A measure of the cost of this search is the number of comparison accesses needed to either find the attribute or determine that it is not present in the record. To minimize the search time the attributes would be maintained in sequence. Either sequential search or binary search would be used. For binary search [21]:

$$E(a_i) = (e_i - 1) + e_i 2^{-e_i+2} \quad ^{e_i+1},$$

where e_i is $\lceil \log_2(m \cdot p_i) \rceil$.

For a sequential search

$$E(a_i) = (1 - \frac{p_i}{2})m \; E(r_{ji}).$$

The optimum search strategy can thus be determined if the relevancy statistics are known.

Given an estimate of the required number of accesses, 1 in the case of the formatted record and the best $E(a_i)$ in the unformatted case, and given the storage requirement estimates for the two record organizations, we can assign costs to the different record alternatives. These costs may then be used to decide between them.

4.1.3 Logical File Organization

RACIS operations and the data base it manages are directed toward satisfying the requirements of its prospective users in government and industry. "These users include research and development activities, design engineers, reliability engineers, specification engineers, quality control engineers, and component part manufacturers. User satisfaction is directly related to the information contained in the data files. The data base must be relevant to the problems encountered and technically credible."[1] Although data is accepted in the form most convenient to the originator, RAC must interpret and reduce the data to a form amenable to search and analysis. Incoming data includes text and numerical parameters and results. The numerical data must be stored in a form such that arithmetic operations and comparisons can be performed. Since the interpretation of the numbers varies from document to document it must be fully identified as to source and type. The bulk of the numerical data describes a test or environment upon a set of microcircuits and the outcome of the process in terms of failure statistics.

The two types of data are processed differently. They are also accessed differently. To meet the differing needs two primary files are required:

a. A document file containing bibliographic information including a brief abstract.

b. A data summary file containing test and environmental failure data for microcircuits.

The two files must contain cross-references to each other to permit tracing data to its sources documents and to permit retrieval of the data in a document in a processable form. The accession number of the document or data summary can be used as link to go from one file to the other. The DM-1 binding capability can be used to combine data items from both files into a single virtual file if required.

4.1.4 Physical File Organization

The use of a specific data management system, DM-1, constrains the choice of file organization techniques. The SERIAL file organization permits the specification of sequential access to a record. RANDOM or RANDOM-SERIAL organizations permit access of the root segment of records via hierarchically structured indexes. Other file access techniques, such as hash coding cannot be employed within the structure of DM-1.

Decisions concerning file access strategies are thus effectively reduced to:

1. The choice between DM-1 physical file types:

   a. RAMDON; b. RANDOM-SERIAL; c. SERIAL

2. The set of fields to be indexed.

In the RAC application the decision rule used for file type was simply expected length of file. Short files such as personal author and failure mode were defined as SERIAL. The longer files were defined as RANDOM. RANDOM files requiring a SERIAL substructure were defined as RANDOM-SERIAL.

Indexing choices are more difficult. The choice should be made on the basis of qualification field frequencies and field discrimination. These figures are, of course, available for only a small sample of queries made under the manual system.

The decision to index a field may be made in terms of the cost model. A data access subsystem S_a may be defined as a system which given a field value v and a state σ as input, yields a pointer p to a record containing v and the same state σ as output, i.e.

$$S_a(v,\sigma) = \langle p,\sigma \rangle$$

This system has many implementations. We will focus on the two resulting from the index/no-index alternatives in a DM-1 RANDOM file. The existence costs for the no-index implementation involves only the costs associated with the code required to initiate a sequential search of an address list and compare the field value against the search argument in each record. For the indexed implementation, existence costs include the costs of creating and maintaining an index as well as the code to access it. Use costs for the no-index case involve the cost of a sequential search of all records on the address list. Use costs in the indexed case include the search through the index. The decision to index should be made if the cost of the indexed access subsystem for the field is less than the cost of the non-indexed implementation. This difference will, in general, be a function of the length of the address list and the number of accesses.

A typical query either contains or may be canonicalized into a union of intersections of attribute-value pairs. For each string of "anded" pairs the addresses of the qualified records may be obtained by intersecting the address lists corresponding to each pair. In the sequence of intersections to be performed the indexed fields are processed first. Thus if the address list is short after processing of the indexed fields, there may be little gain in avoiding a sequential search of the remaining addresses. A field that usually appears with indexed fields may gain little from indexing.

4.2 The RAC Data Base

The value of the data collected and maintained by RAC lies in the extensive collection of microcircuit failure statistics and in the depth of documentation detailing events, conditions and techniques relating to device failures. To perform its role effectively, the RAC Data Base must contain both statistical information and the associated documentation. There must also exist cross-referencing capabilities permitting the retrieval of documents relevant to a set of failure statistics and retrieval of statistics via document references.

Such a data base must be designed to accommodate changes in microcircuit technologies and in the data usage patterns. It must be capable of dealing with new descriptors for components and documents that will arise as new technologies are developed.

To meet the requirements for both failure statistics and documents the proposed data base contains two major files: a Document File and a Data Summary File.

The Document File contains bibliographic data. Subsumed files, whose values are indexed keys to the Document File, include Data Summary Numbers, Part Numbers and Keywords. The Keyword Code File is an auxilliary file containing the display values of keywords and their codes. Only the encoded keywords are stored in the Document File and its indexes.

The Data Summary File contains microcircuit failure data. In addition to fixed sets of device and tests descriptors, subsumed files of miscellaneous descriptors may be used to accommodate the new attributes. An auxilliary Attribute Code File is used to hold display values of the device and test attributes.

Both the Document File and the Data Summary File are random serial with primary key of ACCESSION NUMBER.

Figures 4.3 and 4.4 detail the logical structure of the data base. The codes for the type field and the interpretation of the length field for Figure 4.4 is given below:

TYPE	MEANING	LENGTH UNITS
I	Binary Integer	Bits
H	Alphanumeric Character	Characters
F	Floating Point Number	Words

The organization given implements value codings for some of the Document File bibliographic data and keywords as well as most Data Summary attribute values. Further value coding is possible. The encoding of these attributes' values depends upon the ability to tabulate the domains of possible values for those attributes. They are not currently tabulated hence no encoding could be developed.

The two code files are used to map value codes into display values and vice versa. The relatively large space of values for document keywords, currently 1100 and increasing at the rate of 100 per year, requires a specialized organization for rapid retrieval. The other coded attributes have relatively small value spaces making retrieval efficiency less critical.

The NUMBER OF DOCUMENTS INDEXED field can be used to given an estimate of the value of a keyword either individually or as part of a query. An estimate function F can be defined mapping Boolean combinations of keywords i.e. queries into estimates of the number of documents meeting that query. If K is a keyword then F(K) is the contents of the NUMBER OF DOCUMENTS INDEXED field for K. If Q_1 and Q_2 are queries then:

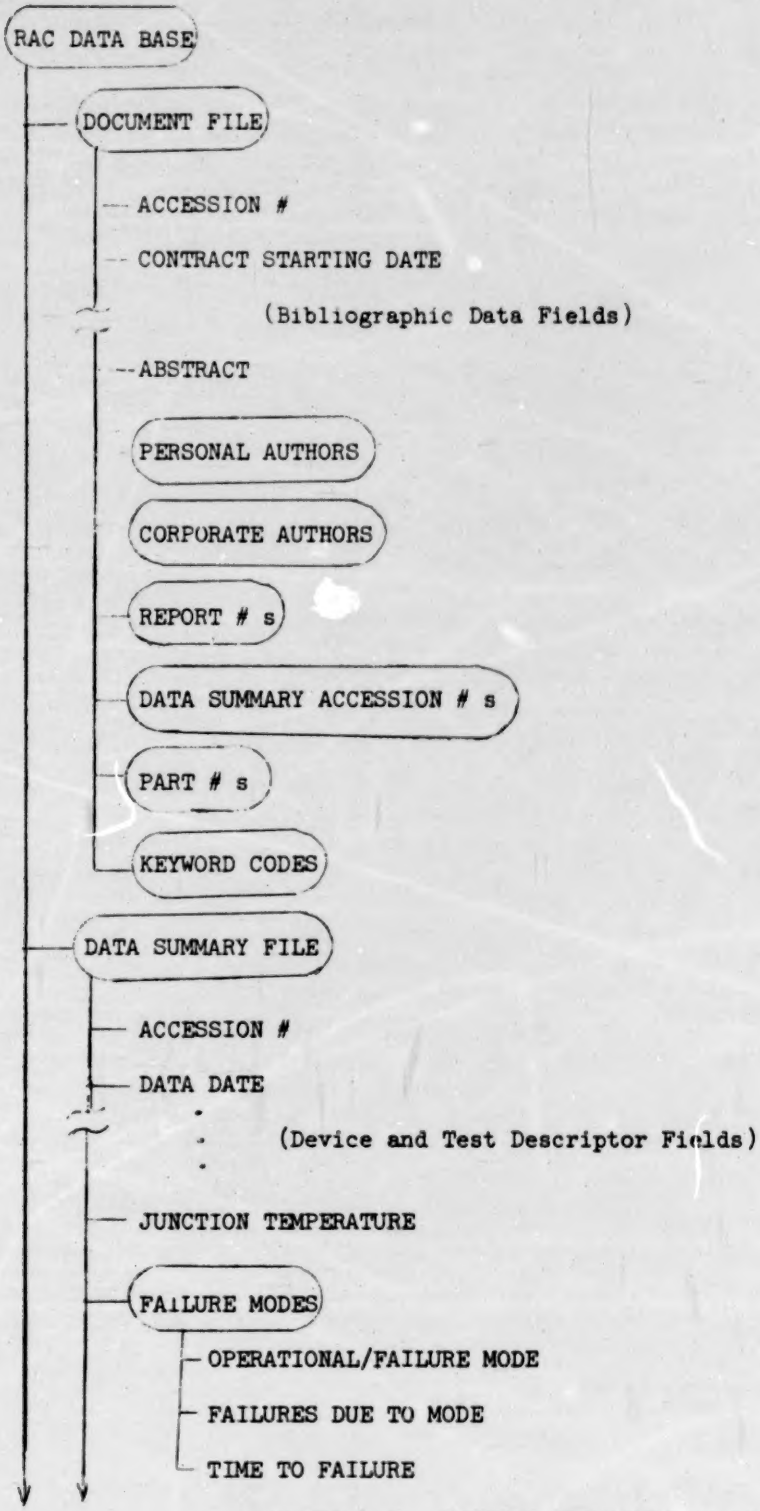


FIGURE 4.3 GENERAL STRUCTURE OF RAC DATA BASE

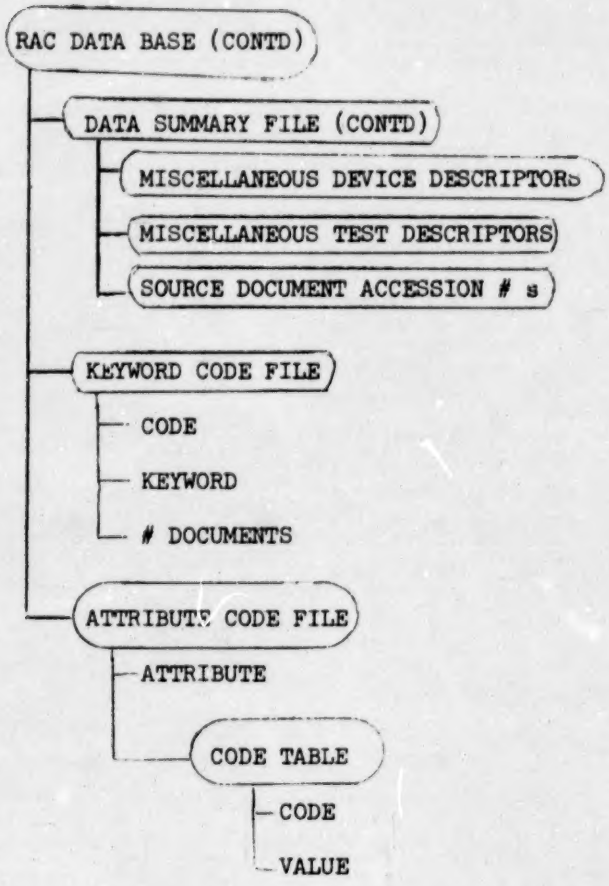


FIGURE 4.3 (CONCLUDED)

FILE OR FIELD NAME	TYPE	LENGTH
RAC DATA BASE	DATA BASE	
DOCUMENT FILE	RAND-SER FILE	
ACCESSION NUMBER	I	36
CONTRACT START DATE	H	6
CONTRACT END DATE	H	6
REPORT DATE	H	6
DOCUMENT ENTRY DATE	H	6
DOCUMENT TYPE	I	8
MEDIA	I	8
SECURITY CLASSIFICATION	I	8
PROPRIETARY	I	1
PAGES	I	36
GOVERNMENT CONTRACT NUMBER	H	32
DOCUMENT TITLE (BEGINNING)	H	52
DOCUMENT TITLE (REST)	H	VAR TO 200
ABSTRACT	H	VAR TO 800
PERSONAL AUTHORS	SER FILE	
PERSONAL AUTHOR	H	52
CORPORATE AUTHORS	SER FILE	
CORPORATE AUTHOR	H	52
REPORT NUMBERS	SER FILE	
REPORT NUMBER	H	32
DATA SUMMARY NUMBERS	SER FILE	
DATA SUMMARY NUMBER	I	36
PART NUMBERS	SER FILE	
PART NUMBER	H	20
KEYWORDS	SER FILE	
KEYWORD CODES	I	36
DATA SUMMARY FILE	RAND-SER FILE	
ACCESSION NUMBER	I	36
DATA DATE	H	6
ENTRY DATE	H	6
BASIC TECHNOLOGY	I	4
SECURITY CLASSIFICATION	I	8
PROPRIETY	I	1
SCREEN CLASS	I	4
OPERATIONAL TYPE	I	3
OPERATIONAL SUBTYPE	I	4
DEVICE FUNCTION	I	5
PACKAGE TYPE	I	2
PACKAGE DETAIL	I	3
METAL SYSTEM	I	4
FIELD INTERCONNECTION	I	3
FIELD SURFACE PASSIVATION	I	2

FIGURE 4.4 DETAILED STRUCTURE OF RAC DATA BASE

FILE OR FIELD NAME	TYPE	LENGTH
DIE BOND	I	3
CRYSTAL PLANE	I	2
DIESIZE	F	1
METAL AREA	F	1
DIFFUSION AREA	F	1
NUMBER OF DIFFUSION STEPS	I	36
IGFET THIN OXIDE AREA	F	1
NUMBER OF TRANSISTORS	I	36
GATE OXIDE	H	8
FIELD OXIDE	H	8
OXYGEN DIFFUSION OVERLAP	F	1
METAL DIELECTRIC THICKNESS	F	1
SUBSTRATE RESISTIVITY	F	1
TEMPERATURE RANGE HIGH	F	1
TEMPERATURE RANGE LOW	F	1
SPEC REFERENCE	H	16
DATA SOURCE (TYPE OF TEST OR ENV)	I	4
TEST TYPE	I	5
TEST SUPPLY VOLTAGE %	F	1
TEST SUPPLY POWER %	F	1
APPLICATION ENVIRONMENT	I	5
EQUIPMENT CLASS	I	4
NUMBER TESTED	I	36
TEST DURATION	F	1
TOTAL PART HOURS	F	1
NUMBER OF FAILURES	I	1
POWER DRAIN %	F	1
THETA SUB JA	F	1
DELTA SUB T	F	1
JUNCTION TEMPERATURE	F	1
FAILURE MODES	SER FILE	
FAILURE MODE	H	32
FAILURE MECHANISM	H	32
FAILURES DUE TO MODE	I	36
TIME TO FAILURE	F	1
MISCELLANEOUS DEVICE DESCRIP.	SER FILE	
MISC. DEVICE DESCRIPTOR	H	8
MISCELLANEOUS TEST DESCRIP.	SER FILE	
MISC. TEST DESCRIPTOR	H	8
SOURCE DOCUMENTS	SER FILE	
ACCESSION NUMBER	I	36

FIGURE 4.4 (CONTINUED)

FILE OR FIELD NAME	TYPE	LENGTH
KEYWORD CODE FILE	RANDOM FILE	
CODE	I	36
KEYWORD	H	16
NUMBER OF DOCUMENTS INDEXED	I	36
ATTRIBUTE CODE FILE	RAND-SER FILE	
ATTRIBUTE	H	18
CODE TABLE	SER FILE	
CODE	I	8
VALUE	H	8

FIGURE 4.4 (CONCLUDED)

$$F(Q_1 \wedge Q_2) = F(Q_1)\ F(Q_2)/N$$

$$F(Q_1 \vee Q_2) = F(Q_1) + F(Q_2)$$

$$F(\neg Q_1) = N - F(Q_1)$$

where N is the number of documents in the file. In an interactive environment this estimate can be made during the query processing phase and displayed to the inquirer before the actual document search begins. Queries leading to very large or very small estimates can be rephrased.

A great deal of the file space will be occupied by document abstracts. In the current system they average 100 English words in length with some containing as many as 550 words. There is some question regarding the justification for keeping abstracts of this length on random access storage. The requirement for lengthy abstracts can probably be met through some combination of microfilm and off-line storage on tape or mountable disks.

The basic design structure of RACIS involves the way data is represented in the system and how this representation is communicated with the user community. The code files help to make user interface programs insensitive to the data base. The tables can be used as an edit check for incoming data or as a response prompter for interactive input. They can also be used to make the output of the system much more readable and understandable.

4.3 System Functions

In addition to defining data base organization, the objective of this study, other tasks, must be performed before the RACIS is operational. One of the major tasks is the development of the capability for performing file maintenance and file access tasks. Some of these tasks are:

a. File creation. This task need be executed only once. It requires that the input data be available in a machine readable form. It uses the DM-1 data base definition to convert the data into the data base internal format and build the data base.

b. File purge. This task will be run annually to remove obsolete data from the data base.

c. Data entry. This task will be an on-going activity. Human factors considerations must be paramount, communication between system and user must be both efficient and natural to maximize productivity and minimize error rates. Since errors are inevitable, error correction capability must be a concommitant to data entry.

d. File modification. Modification is required to correct errors and to add or modify document or data summary descriptions in light of new information.

e. Response to queries. Response time and user convenience are the paramount considerations for this frequently performed task. User requests for specific data and for the opportunity to browse through data must be accommodated. While some responses must b e in hard copy form printed output is not required for data used in browsing.

f. Report generation. The ability to generate RAC current publications automatically as well as other publications and tables requires a report formatting and generation capability. This capability involves:

(i) Selection of required information from the data base.

(ii) Processing the data (see g below)

(iii) Sorting

(iv) Formatting the data for output.

g. Complex processing. Statistical analysis, simulation regression etc. to require the development of a library of programs capable of operating with data extracted from the data base. Mechanisms for specifying the selection of data to be bound to the program are available in RACIS through the DM-1 binding job. A determination of the requirement for immediate response processing as an extension to simple file queries is required.

4.4 System Implementation

The implementation of the RAC/DM-1 data base requires careful planning and monitoring to assure an effective product. Some of the steps required are given in Figure 4.5 and are detailed below:

A receiptive environment for the data base must be built. This involves the definition of data base structures and formats. It also requires the selection of the host computer and auxilliary storage media where the data base is to reside. The storage required must be allocated to the data base.

The task of transforming the data from its current form to that required must be defined. This involves defining the input medium to be used and input formats and procedures. This is the best time to stop error propagation therefore any edits or input validation must be defined at this point, as well as the measures to be taken to correct errors. Any encoding or decoding of data must be specified. To monitor the transformation process, listings of data added to the data base should be specified.

The actual transformation of the data to machine readable form involves the selection of a data entry service e.g. keypunch or key to tape or disk. To assure success, samples of data entry output can be tested against the edits and data transformation defined earlier. After making any modifications to data entry instructions, media formats or transformation routines, the rest of the data may be keyed subject to periodic validation of samples. The procedures used in the initial construction of the data base need not be the same ones used in operational input of data. Once operational, a more interactive data entry procedure may be desired.

1. Preparation of a receptive environment

   1.1 Define data base structure and formats

   1.2 Define computer and storage device where data base is to reside.

   1.3 Allocate direct access storage.

2. Preparation of the data base creation transformation

   2.1 Define input medium

   2.2 Define input formats and procedures

   2.3 Define any edits or validation of input required

      2.3.1 Define responses to invalid data

   2.4 Define any required transformations of input

   2.5 Define outputs required for further validation

   2.6 Define error correction procedures

3. Transformation of data to machine readable form

   3.1 Select keypunch or data entry service

   3.2 Test sample output in format selected in 2.2 against procedures defined in 2.3, 2.4 and 2.5

      3.2.1 Make required revisions in format or procedure

   3.3 Data Entry Service completes transformation

      3.3.1 Periodic samples are validated as in 4.

4. Data Base Validation and Error-Correction

   4.1 People knowledgeable of data base contents examine samples of data base as generated.

   4.2 Errors corrected using procedures in 2.6

FIGURE 4.5 STEPS IN THE IMPLEMENTATION OF RAC/DM-1 DATA BASE

5. DISCUSSION

One of the basic purposes in system design is to decide which changes should be accommodated with what degree of ease. It is therefore, important to anticipate the degree of change expected in each of the major components of a design and, where flexibility can be introduced without excessive cost, design for change. Responsiveness to change is one of the primary reasons for using a GDMS. It compartmentalizes the effects of change by isolating programs from the data base and hence from other programs as well. IR systems are designed to accommodate a large, growing and partially structured attribute space. They must deal with new concepts in an orderly way. Combined GDMS/IR systems present an opportunity to develop a collection of programs useable against a rapidly changing data environment.

In this study we have defined a cost model that can be used in decision making that is system design. We have applied this criterion to a set of design problems and applied the solutions obtained to a proposed data base organization for RAC.

Generalized data management systems have been promoted as a method for dealing with data that crosses functional and application lines. Its effectiveness in this role is as a common set of data handling routines and data descriptions usable by a wide variety of application programs. The problems of technical information retrieval centers fall into this category. The single purpose document retrieval system is inadequate to cope with the demands placed upon it in this environment. The GDMS can obviate the need for cumbersome interfaces between document using programs and the document retrieval system.

BIBLIOGRAPHY

1. Auerbach Corporation. Reliability Analysis Center Data Processing System Design Specification Report. New York: Rome Air Development Center, TR 66-474, August, 1966.

2. Auerbach Corporation. Addendum to Functional Specification DM-1/GE-635. New York: Rome Air Development Center, TR-69-102, May, 1970.

3. Benner, F.H. On Designing Generalized File Records for Management Information Systems. Proceedings of AFIPS, Fall Joint Computer Conference, 1967, p. 291.

4. Capraro, G.T. Proposal for RAC/DM-1, 1971.

5. Cargaro, A. et al. Addendum to Rome Air Development Center TR-66-474, Functional Specification DM-1/GE-635. New York: Rome Air Development Center, TR-69-102, June, 1969.

6. Castleman, P.A. "Data Management Techniques for the Social Sciences". Data Bases, Computers and the Social Sciences, pp. 139-157, Bisco, R.L. ed. Wiley-Interscience, New York, 1970.

7. Codasyl. Introduction to Feature Analysis of Generalized Data Base Management Systems. Codasyl Systems Committee, Communications of the Association for Computing Machinery, Volume 14 #5, May, 1971, p. 33.

8. Codasyl. Data Base Task Group Report., New York: Association for Computing Machinery, October, 1969.

9. Codasyl Systems Committee. A Survey of Generalized Data Base Management Systems. New York: Association for Computing Machinery, May, 1969.

10. Codasyl Systems Committee. Feature Analysis of Generalized Data Base Management Systems. New York: Association for Computing Machinery, May, 1971.

11. Edwards, J.S. Adaptive Man-Machine Interaction in Information Retrieval. Pennsylvania: University of Pennsylvania, 1967. Ph.D. Thesis.

12. Humes, J.W. A Linear Model for the Evaluation of User-Interactive Retrieval Systems. Pennsylvania: Lehigh University, 1969. Ph.D. Thesis.

13. Jackson, D.M. Classification, Relevance and Retrieval. Advances in Computers, Vol. 11, 1971, pp. 60-122.

14. Jacobi, Lauffenburger, et al. Development and Implementation of a Reliability Analysis Center. New York: Rome Air Development Center, TR-68-339, October, 1968.

15. Lauffenburger, H.A. Operation of Reliability Center. Rome Air Development Center, TR-69-360, TR 71-7, January, 1970 and January, 1971.

16. McCusky, W.A. "On Automatic Design of Data Organization", Proc. FJCC, 1970, pp. 187-199.

17. Meadow, C.T., "The Analysis of Information Systems: A Programmer's Introduction to Information Retrieval", Wiley, New York, 1967.

18. Minker, Jack. General Data Management Systems - Some Perspectives. University of Maryland, Computer Science Center, December, 1969, TR 69-100.

19. Nunamaker, J.F. On the Design and Optimization of Information Processing Systems. Ohio: Case Western Reserve University, 1969.

20. Pan, George Shing-Tan. Design of an Information System Data Base. New York: Syracuse University, 1967.

21. Price, C.E. Table Lookup Techniques. Association for Computing Machinery, Vol. 3, #4, June, 1971., p. 49.

22. Senko, M.E. "Information Storage and Retrieval Systems", Advances in Information Systems Science, Tou, J.T. ed., pp. 229-282, 1969.

23. Watson, Robert D. A Systems Approach to Investigate the Optimum Cost Effectiveness of a Generalized Management Information System. Oklahoma: University of Oklahoma School of Industrial Engineering, July 1, 1970.

24. Williams, J.H. "Computer Classification of Documents", Mechanized Information Storage, Retrieval and Dissemination. Proceedings of F.I.D./IFIP Conference, Rome, June, 1967. Samuelson, K. ed. North Holland Publishing Co. Amsterdam 1068.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-73-410	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Data Management Systems for Structured Information Retrieval		5. TYPE OF REPORT & PERIOD COVERED Interim (July 71 to July 72) 6. PERFORMING ORG. REPORT NUMBER Technical Report No. 72-3
7. AUTHOR(s) Mr. Leo H. Groner Mr. Amrit L. Goel		8. CONTRACT OR GRANT NUMBER(s) F30602-72-C-0281
9. PERFORMING ORGANIZATION NAME AND ADDRESS Syracuse University Syracuse, New York 13210		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 55810213
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (ISIM) Griffiss AFB, NY 13441		12. REPORT DATE February 1974 13. NUMBER OF PAGES 45
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release. Distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Rocco F. Iuorno (ISIM)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Data Processing
Digital Computer
Data Management System
Information Retrieval

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes some of the advantages and problems of using generalized data base management systems in information retrieval. A particular system, Data Manager -1, was studied to derive use and cost criteria. System costs were then obtained as a function of cost components and user demands for outputs. The methodology developed has been applied to a set of design problems relevant to GDMS/IR data base design.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

BEST COPY AVAILABLE

MISSION
of
Rome Air Development Center

RADC is the principal AFSC organization charged with planning and executing the USAF exploratory and advanced development programs for electromagnetic intelligence techniques, reliability and compatibility techniques for electronic systems, electromagnetic transmission and reception, ground based surveillance, ground communications, information displays and information processing. This Center provides technical or management assistance in support of studies, analyses, development planning activities, acquisition, test, evaluation, modification, and operation of aerospace systems and related equipment.